Lab 7
November 28, 2017
Multirate Signal Processing and Image Filtering

INSTRUCTIONS:
All lab submissions include a written report and source code in the form of an m-file. The report contains all plots, images, and figures specified within the lab. All figures should be labeled appropriately. Answers to questions given in the lab document should be answered in the written report. ***The written report must be in PDF format***. Submissions are done electronically through **Compass 2g**.

If you're not that experienced with up and down sampling, you may want to do the problems in the following order 4,5,6,7,2,1,3.

1 Discussion

Sample rate conversion is, by definition, changing the sample rate of a discrete signal. Upsampling is the addition of samples to a signal to effectively increase the sample rate of the signal. Conversely, downsampling is the decimation of a signal to effectively decrease the sample rate. In either case, the total length of the signal (in time) does not change. When the number of samples increases, the sample rate decreases such that the total time remains the same. For example, music can be sampled at a low bitrate is the same length as music sampled at a high bitrate. A higher bitrate just corresponds to more samples per second which corresponds to higher frequencies captured.

2 Upsampling and Downsampling

A discrete signal must be band limited in order for the signal to be upsampled. A signal which is not bandlimited can still be upsampled, but the results will be undesirable. Upsampling is a 2 step process. For example, upsampling by U corresponds to inserting $U-1$ zeros between each sample of the original signal. More precisely, given that $x(n)$ is the original signal with length N and $x_u(n)$ is the upsampled signal,

$$x_u(m) = \begin{cases} x(n) & m = Un \\ 0 & else \end{cases} \tag{1}$$

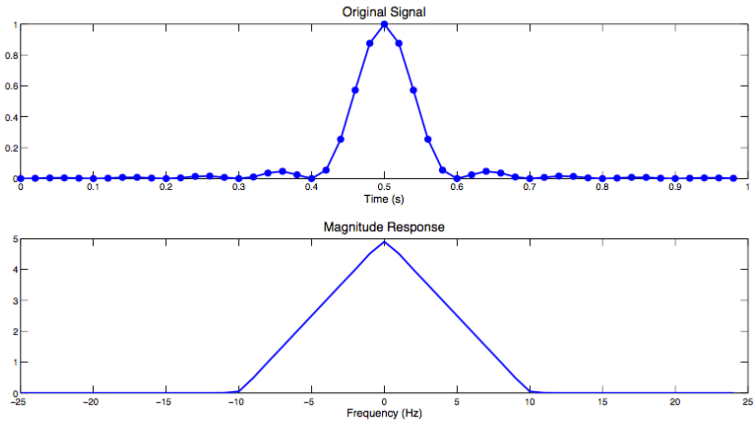

Figure 1: The original signal $x(n)$ and its magnitude response

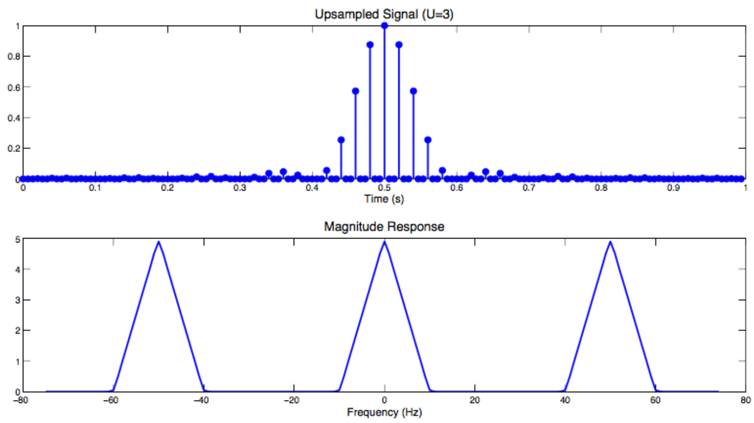

Figure 2: $x(n)$ upsampled by $U = 3$. This increases the sampling frequency by 3 and also introduces copies in the frequency domain due to the scrunching of the frequency axis.

The insertion of $U-1$ zeros between samples of $x(n)$ causes the frequency axis to scrunch by a factor of U. However, this scrunching is only apparent in the DTFT, where $-\pi < \omega < \pi$. In the Ω domain, there is no scrunching. That's because $\Omega = \omega \cdot f'_s$ where $f'_s = f_s U$. So the scrunching in the ω domain comes from the fact that f'_s for $x_u(n)$ is larger than f_s. Pay special attention to the time axis of each graph. The duration of the signal from a time perspective does not change.

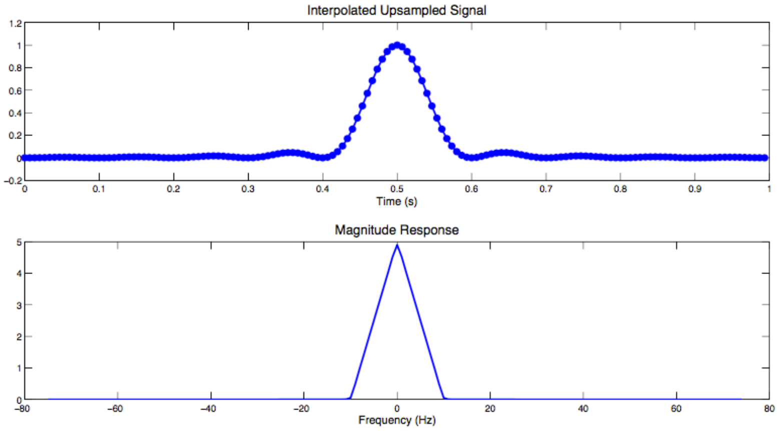

Figure 3: Applying an ideal LPF in the frequency domain to remove the copies.

The spectrum of $x_u(n)$ has the same shape as the spectrum of $x(n)$, barring the additional copies. To remove the additional copies, we can apply an ideal LPF. In MATLAB, this can be done by letting $X_u(\omega) = 0$ for $|\omega| > f_s/2$ where $X_u(\omega)$ is the DTFT of x(n). The result is shown in Figure 3. Figure 4 superimposes $x(t)$ and $x_u(t)$ to show that one is the upsampled version of the other.

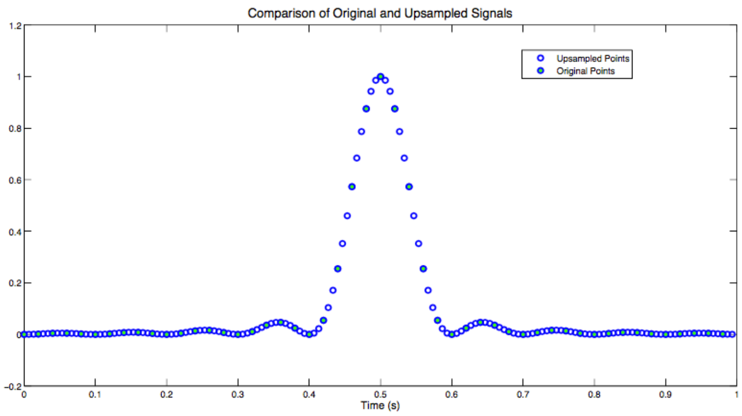

Figure 4: Comparison of the original signal $x(n)$ to its upsampled version $x_u(n)$.

By comparison, downsampling is much simpler than upsampling. The downsampling operation can be expressed as

$$x_d(n) = x(nD) \tag{2}$$

where D is an integer greater than 1. The choice of D is restricted by Nyquist sampling rate, which can be violated if D is chosen to be too large. Upsampling and downsampling can be combined to achieve the desired sample rate. The final sample rate can be expressed as $f_s' = \frac{U}{D} f_s$.

Report Item 1: An audio technician accidentally set the sample rate to 66,150 Hz during an important recording session. This is 1.5 times the standard 44,100 Hz. Having already said goodbye to the musicians, he happens to come across an ECE311 student who understands sample rate conversion. Read the sound **audioclip.wav** using **audioread**. What values of U and D can you choose such that the final sample rate is 44,100 Hz. Plot the magnitude spectrum (in dB) of the upsampled signal before applying the lowpass filter, after applying the lowpass filter, and post downsampling. Be sure to scale the frequency axis Ω **in Hz** with respect to the proper sampling frequency. Use **sound** to play the original sound and the sound after sample rate conversion **at the new sampling rate, 44,100 Hz**. Does the original sound faster or slower compared to the new sound?

Report Item 2: Luckily, the audio technician's dad owns the record company, and therefore he did not lose his job. Not too long after, the same audio technician was asked to record the newest electronic music album "The DSPain for life" from Ben and Yu-Jeh. While he was eating nachos during the studio sessions, he opened up the door to catch some fresh air. Surprisingly, Bruce Wayne lives just nearby, and one of the bats was curious enough to leave the bat cave and flew right into the studio without anyone noticing. Bats are known to communicate with ultra-high frequency sound waves, and so the hit song from Ben and Yu-Jeh was corrupted.

(a) Load **song1_corrupt.mat** and plot the FFT frequency response. Here we have the song recorded at a sampling rate of 44100 Hz. Can you tell which could be the bat's sound ? What is the frequency for that ?

(b) Listen to it. Can you hear the bat? Why or why not? **HINT:** human hearing has a typical lower bound and upper bound for the frequency of a sound.

(c) Without a LPF, downsample the song by $D = 2$. Now, you have the song with a sampling rate of 22050 Hz. Can you now hear the bat? Plot the FFT frequency response and explain.

(d) Now, we know that without a LPF, the downsampling could cause some problem. Look at the frequency response obtained in part (a) and design an LPF for downsampling by $D = 2$

(e) Apply the LPF designed in part (d) and downsample the original corrupted sound signal by $D = 2$. Can you hear the bat? Plot the FFT frequency response and explain.

3 Spotify Data Transfer Issue

You're a Graduate of the ECE program from the University of Illinois, working as an audio engineer at Spotify. On March 10, 2019, a hit song (to be remain unnamed), goes on the radio. It goes viral. Suddenly, at the Spotify Server center, they realize that the data transfer rate has reached its limit. Your boss, Mr. Kuo, believes that you can downsample the data and have the software at the receiver end do upsampling and interpolation. This may fix the data transfer issue.

Report Item 3:
Load the file 'songz.mat', a variable named `good_news` should appear on the screen. This contains a song that has not been upsampled or downsampled; it's the original. The sampling rate for this song is 48000 Hz. Determine an acceptable upsampling and downsampling rate for this problem. At the minimum, include a plot of the magnitude spectrum of the downsampled spectra, upsampled spectra, the magnitude response of your interpolation filter, and the magnitude response of your filtered signal. Listen to your final song to ensure if the quality is acceptable. Briefly explain the process of determining these rates.

Yep, we didn't give you a lot of information about this problem. In the real engineering world, you many times don't get all the details and specifications about a problem. If there are questions, by all means, ask a TA or post on Piazza. We'll do our best to answer.

4 2-D DFT

The definition of 2-D DFT is

$$X(u,v) = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} x(m,n) e^{-i2\pi(mu/M + nv/N)} \tag{3}$$

Like the 1-D DFT, the 2-D DFT uses sinusoidal basis functions except they're in 2-D. In 1-D, we're used to dealing with time and frequency. 2-D DFT is often used for image processing where we deal with space and, correspondingly, spatial frequency. Assume that $x(m,n)$ represents an $M \times N$ image where $m = 0, \ldots, M-1$ and $n = 0, \ldots, N-1$. Then u indexes variations in the vertical direction and v indexes variations in the horizontal direction. Physics students may be more familiar with the wavenumber notation

$$k_x = 2\pi v/N \tag{4}$$

$$k_y = 2\pi u/M \tag{5}$$

where k_x represents variations in the horizontal direction and k_y represents variations in the vertical direction. The 2-D DFT can be re-written as

$$X(u,v) = \sum_{m=0}^{M-1} e^{-i2\pi mu/M} \sum_{n=0}^{N-1} x(m,n) e^{-i2\pi nv/N} \tag{6}$$

which can be implemented using a series of FFTs. The number of operations isapproximately $\mathcal{O}(MNlogN) + \mathcal{O}(NMlogM)$ or $\mathcal{O}(NMlogNM)$.

Report Item 4: Create a function called *myDFT2* that implements the 2-D DFT using the 1-D FFT. Verify your results using **fft2**.

$$f(m,n) = cos(k_y m + k_x n)$$

Then implement $f(m,n)$ using **meshgrid**. Use **imagesc** to plot $f(m,n)$ for $(k_x, k_y) = (\pi/4, 0)$ and its 2-D DFT(magnitude only). Repeat for$(k_x, k_y) = (0, \pi/4), (k_x, k_y) = (\pi/4, \pi/4), and (k_x, k_y) = (\pi/4, -\pi/4)$. Make sure to label both k_x and k_y axes.

One should notice that, like 1-D Fourier transforms, the 2-D Fourier transform increases in frequency further away from the origin. However, there is an additional component to 2-D Fourier transforms not seen in 1-D: the idea of rotation. So not only do we have magnitude, phase, and frequency, we also have rotation angle.

5 Image Filtering

An image can be filtered using 2-D convolution, which can be expressed as

$$Y(m,n) = \sum_{m'}\sum_{n'} X(m',n')h(m-m',n-n') \tag{7}$$

where $h(m,n)$ is the filtering kernel. An example of a low-pass filter is

$$h(m,n) = \begin{bmatrix} 1/8 & 1/16 & 1/8 \\ 1/16 & 1/4 & 1/16 \\ 1/8 & 1/16 & 1/8 \end{bmatrix} \tag{8}$$

There are several ways to recognize that this is a lowpass filter. First, using the fact that the DC component of the Fourier transform is proportional to the sum of the coefficients in $h(m,n)$, it can be seen that $\sum_m \sum_n h(m,n) = 1$. Since the DC component is non-zero, it is likely a lowpass lter. Another clue is the fact that the filter looks Gaussian. Since a Gaussian in the spatial domain is also a Gaussian in the frequency domain, the filter is a lowpass.

Report Item 5: Load *image1.jpg* using **imread**. This image contains what is known as salt and pepper noise. Filter this image using the lowpass filter given in (8) and plot the result using **imagesc**. Use **conv2** to apply the filter.

In contrast, the Laplacian filter is a highpass filter. This can be deduced from the fact that the sum of its coefficients equals zero, which suggests that it cannot be a lowpass. A highpass filter can be used for edge detection in an image.

$$h(m,n) = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \qquad (\textit{Laplacian Filter}) \tag{9}$$

Report Item 6: Load *image2.jpg* using **imread**. Apply the Laplacian filter and plot the resulting figure using **imagesc** with **colormap('gray')**. Use **conv2** to apply the filter.

6 Listening to a Picture, Looking at Sound

A Professor I had once (who shall remain unnamed), asked me what one of my images sounded like. I was baffled by the question. It took me, maybe 3 years, to finally understand what he/she was saying. We'll ask you the same here.

Report Item 7:

1.) Generate a DFT matrix of dimension 100×100. Store the real part of the matrix in one variable and store the imaginary part in another.
2.) Generate a convolution matrix of a filter [1,2,3,4,5,6]. Suppose your input signal is of length 50.
3.) Generate a 'motion' filter using the fspecial function in Matlab, set $\theta = 30$ degrees and a length of 50.
4.) Plot these 4 matrices using imagesc in a four panel plot.
5.) Vectorize each of these matrices, and describe what you hear.
6.) Do you hear anything interesting about the real and imaginary DFT matrix? Use sound with a f_s of 44100 Hz.
7.) Philosophically, what is your interpretation of the statement "Listen to a Picture, Look at a Sound." (One sentence is sufficient, do not spend more than 2 minutes on this part, write from the heart.)